**Министерство науки и высшего образования**

**Российской Федерации**

**ФЕДЕРАЛЬНОЕ Государственное Автономное**

**образовательное учреждение высшего образования**

**«новосибирский национальный исследовательский государственный университет»**

**Факультет информационных технологий**

Кафедра общей информатики

Направление подготовки 09.03.01 Информатика и вычислительная техника

Направленность (профиль) Программная инженерия и компьютерные науки

**ОТЧЕТ**

**о прохождении учебной практики (научно-исследовательской работы (получение первичных навыков научно-исследовательской работы))**

(указывается наименование практики)

**Обучающейся** Шипаевой Марии Алексеевны **группы №** 22201 **курса** 3

(Ф.И.О. полностью)

**Тема задания**: Разработка алгоритма хирального анализа асимметрии для исследований лицевых выражений.

**Место прохождения практики:** ФГБНУ "Научно-исследовательский институт нейронаук и медицины", Лаборатория дифференциальной психофизиологии, 630117, Новосибирская обл., Новосибирск, ул. Тимакова, 4

(полное наименование организации и структурного подразделения, индекс, адрес)

**Сроки прохождения практики:** с 07.10.2024 г. по 23.12.2024 г.

**Руководитель практики   
от профильной организации** Савостьянов Александр Николаевич, в.н.с.

(Ф.И.О. полностью, должность) (подпись)

**Руководитель практики от НГУ** Савостьянов Александр Николаевич, профессор

(Ф.И.О. полностью, должность) (подпись)

**Руководитель ВКР** Савостьянов Александр Николаевич профессор

(Ф.И.О. полностью) (должность)

**Оценка по итогам защиты отчета:**

(неудовлетворительно, удовлетворительно, хорошо, отлично)

**Отчет заслушан на заседании кафедры** общей информатики

(наименование кафедры)

**протокол \_\_\_\_\_\_\_\_\_от** «\_\_\_\_\_\_» декабря 2024 г.

Новосибирск 2024

**Содержание**

Введение………………………………………………………………………………………..3

Отчет……………………………………………………………………………………………5

Заключение…………………………………………………………………………………….8

Список литературы……………………………………………………………………………9

Приложения…………………………………………………………………………………10

**Введение**

**Актуальность темы задания:**

Актуальность данной работы обусловлена возрастающим интересом к изучению лицевой асимметрии как потенциального маркера различных неврологических и психических заболеваний. Разработка автоматизированных методов анализа асимметрии, основанных на современных методах компьютерного зрения, позволяет проводить более объективные и точные исследования, способствуя более глубокому пониманию патогенеза этих заболеваний и разработке новых диагностических инструментов. Автоматизация процесса анализа также существенно сокращает время и трудозатраты исследователей, открывая новые возможности для масштабных популяционных исследований.

**Цели:**

1. Разработать алгоритма хирального анализа асимметрии для исследований лицевых выражений
2. Научиться использовать библиотеку PyFeat

**Задачи:**

1. Изучить библиотеку PyFeat, предназначенную для исследований выражения лица
2. Используя библиотеку PyFeat, разработать и написать алгоритм, который бы выравнивал фотографию лица по углу наклона, анализировал его AUs (action units)[[1]](#footnote-1), эмоции и т.д., отзеркаливал оригинальное изображение по средней линии лица и проделывал бы тот же анализ для двух получившихся отзеркаленных изображений
3. Расширить алгоритм для применения его к видео
4. Собрать алгоритм в скрипт для дальнейшего использования

**Место прохождения учебной практики:**

Федеральное государственное бюджетное научное учреждение "Научно-исследовательский институт нейронаук и медицины" (ФГБНУ "НИИ Нейронаук и медицины") в Новосибирске является ведущим российским институтом, специализирующимся на изучении мозга и нервной системы, а также на разработке новых методов диагностики и лечения неврологических и психических заболеваний. Институт проводит фундаментальные и прикладные исследования в области нейробиологии, нейрофизиологии, нейрохимии, нейрогенетики, нейрофармакологии и клинической неврологии.

Лаборатория дифференциальной психофизиологии, как структурное подразделение ФГБНУ "НИИ Нейронаук и медицины", занимается изучением индивидуальных различий в функционировании нервной системы и психики человека. Основные функции лаборатории включают:

* Исследование физиологических основ индивидуальных различий в темпераменте, когнитивных способностях, эмоциональной сфере.
* Разработка и применение методов психофизиологической диагностики для оценки функционального состояния нервной системы.
* Изучение влияния различных факторов (генетических, средовых, социальных) на формирование индивидуальных особенностей психофизиологических параметров.
* Разработка методов коррекции и оптимизации функционального состояния нервной системы с учетом индивидуальных особенностей.

**Предполагаемые результаты прохождения практики:**

Предполагается, что разработанный алгоритм будет корректно анализировать лицевую активность и будет использоваться в дальнейших исследованиях

**Отчет**

1. Использованные методы
   1. PyFeat

Главной моей задачей было разработать алгоритм хирального анализа асимметрии для мордографических исследований на основе библиотеки PyFeat. Эта библиотека, написанная на языке программирования Python, предназначена для анализа и обработки данных о лицевых выражениях и движениях. Весь ее код доступен в открытом доступе на [GitHub](#py_feat_github). Она разработана для работы с результатами, полученными с помощью систем отслеживания лицевых действий (таких как FACS — Facial Action Coding System), и предоставляет инструменты для извлечения различных признаков с лица, таких как ключевые точки лица (landmarks), emotion descriptors (описания эмоций) и Action Units (AUs), что позволяет оценивать эмоциональные состояния и мимику.

Landmarks – это набор точек, которые отмечают ключевые особенности лица, такие как уголки глаз, кончик носа, края губ и т.д. Расстояния и углы между точками могут быть использованы для извлечения геометрических признаков лица, которые характеризуют его форму и пропорции, для анализа движений губ, бровей и т.д. В моем случае я использовала набор из 68 точек (Приложение 1) для определения угла наклона и средней линии лица.

Action units (AUs) – это система кодирования движений лицевых мышц, разработанная Полом Экманом и Уоллесом Фризеном. Каждая AU соответствует конкретному движению одной или нескольких лицевых мышц. Так же определенные комбинации AUs описывают основные эмоции, такие как радость, грусть, гнев, страх и т. д., что позволяет объективно классифицировать лицевые выражений. В контексте хирального анализа сравнение интенсивности одних и тех же AU на левой и правой половинах лица может позволить выявить скрытые паттерны асимметрии, которые могут быть связаны с различными неврологическими или психическими состояниями. PyFeat использует набор из 20 AUs (их описание и расположение можно посмотреть в приложении 2 и 3).

Основной класс, который я использовала, Detector, отвечает за извлечение ключевых точек, AU, эмоций, позы лица в пространстве и идентичности. Метод Detector.detect() принимает на вход строки с путями к файлам или тензоры и возвращает объединенные в pd.DataFrame результаты для всех изображений.

Модели, используемые в PyFeat для обнаружения:

* Ключевых точек: mobilefacenet, mobilenet, mobilenet
* AUs: XGBClassifier, SVMClassifier
* Эмоций: ResMasking, EmoSVMClassifier
* Лиц и поз: resnet\_fpn\_backbone, Faster R-CNN with ResNet18 backbone
* Для распознавания лиц: InceptionResnetV1
  1. PyTorch

Для удобства анализа видео в программе используется библиотека PyTorch. PyTorch - это библиотека машинного обучения с открытым исходным кодом, разработанная Facebook's AI Research lab (FAIR). Она предоставляет широкий набор инструментов и функций для создания и обучения нейронных сетей, а также проведения исследований в области глубокого обучения. Ознакомиться с основами их использования можно на [сайте PyTorch](#py_torch) в разделе с уроками и руководством. В частности, я использовала тензоры PyTorch, с которыми отлично работает PyFeat. Тензоры - это фундаментальные структуры данных в PyTorch. Они представляют собой многомерные массивы, аналогичные массивам NumPy. Их спользование позволяет эффективно использовать возможности GPU для ускорения вычислений, что особенно важно при обработке видео, где требуется анализировать большое количество кадров*.*

Преобразование входных данных в тензоры PyTorch происходит перед вызовом Detector.detect(). Это обеспечивает совместимость с PyFeat и позволяет использовать преимущества тензоров, такие как векторизация и возможность использования GPU. После вызова Detector.detect\_faces(), полученные результаты, включая обнаруженные лица, также представляются в виде тензоров. Это позволяет продолжить обработку в PyTorch, например, выполнить поворот и отзеркаливание лиц.

* 1. OpenCV

OpenCv (Open Source Computer Vision Library или cv2) – это библиотека, предоставляющая мощный инструментарий для решения задач компьютерного зрения и обработки изображений, с которым можно ознакомиться на [сайте OpenCV](#opencv). Разобраться некоторых вопросах мне так же помогли статьи с [Habr](#habr). В моем случае я использую OpenCV для захвата видеоданных из файла с помощью функции cv2.VideoCapture() и преобразования каждого кадра в формат, подходящий для обработки с помощью PyTorch

* 1. Csv

Библиотека csv в Python предоставляет функциональность для работы с файлами в формате CSV (Comma Separated Values), который является одним из самых распространенных форматов для хранения и обмена табличными данными. Библиотека позволяет читать csv файлы, парсить, записывать новые данные и всячески форматировать их. В моем случае csv используется для сохранения результатов анализа видео, что позволяет хранить полученные данные в структурированном, формате, удобном для дальнейшего анализа и обработки.

1. Логика алгоритма для обработки одного лицевого изображения

Алгоритм для обработки изображения состоит из нескольких шагов:

1. На вход подается изображение с лицом человека в анфас. Прежде, чем приступить к анализу асимметрии лица и созданию зеркальных химер, необходимо выровнять оригинальное изображение по углу наклона относительно оси Y. Для этого сначала применяется метод Detector.detect() основного класса Detector библиотеки PyFeat. Результаты, возвращаемые методом (в формате FEX), включают в себя положение головы в пространстве, ключевые точки лица, AUs, эмоции и т.д.

Используя координаты ключевых точек, вычисляем угол наклона головы относительно оси Y. Нам понадобятся три горизонтали лица: первая линия, соединяющая уголки глаз (точки 40 и 43), вторая – соединяющая точки на крыльях носа (точки 32 и 36), третья – соединяющая уголки губ (точки 49 и 55). Для каждой пары вычисляется угол наклона и берется среднее значение.

До использования такого метода я так же пробовала брать все точки глаз и вычислять угол по линии, соединяющей средние точки левого и правого глаза. Данный подход оказался менее эффективным, так как внешние уголки глаз могут быть ассиметричными и портить усредненные значения.

1. Поворачиваем изображение, тем самым выравнивая угол наклона лица. Это необходимо для корректного построения зеркальных химер и их дальнейшего анализа.
2. После выравнивания изображения его необходимо снова проанализировать, чтобы уточнить ранее полученные результаты. Используя новые координаты ключевых точек, находим среднюю линию лица. Изначально я пробовала проводить среднюю линию из точки на кончике носа (точка 34), но такой метод может давать искаженные результаты, так как кончик носа может быть повернут в сторону. Поэтому в итоге я беру центральную точку подбородка и верхнюю точку на перегородке носа (точки 9 и 28), вычисляем среднее значение и, проводя перпендикуляр, отмечаем среднюю линию.
3. Делим изображение по средней линии. Каждую из получившихся половин лица зеркально отражаем, достраивая таким образом их до двух химер исходного лица.
4. Получившиеся химеры аналогично анализируем с помощью метода Detector.detect().
5. Обработка видео

Для того, чтобы проанализировать видео, программа в цикле проходится по всем кадрам и применяет данный алгоритм для каждого из них. На вход подается путь к папке на компьютере. Программа рекурсивно сканирует ее, находя в ней все mp4 файлы. Далее создается файл таблицы с тем же названием, что исходное видео, но с другим расширением. Если такая таблица уже существует, видео пропускается и сообщает об этом пользователю

Порядок действий для анализа видео:

1. Кадр преобразуется в PyTorch тензор. Это делается с целью улучшения производительности и сохранения памяти: если бы программа сохраняла кадр на диске, это бы существенно замедлило обработку видео. Использование тензоров позволяет передавать изображение напрямую в метод Detector.detect() вместо пути к файлу изображения.
2. Применяется алгоритм обработки изображения, только теперь для тензора. Если в кадре было обнаружено несколько лиц, этот кадр не анализируется и пропускается, так как точно нельзя определить, какое именно лицо будет выбрано для анализа.
3. На выходе получаем результаты анализа исходного кадра и двух отражений, которые записываются в csv-файл

Csv-файл содержит следующие заголовки: номер кадра, таймкод, количество обнаруженных лиц в кадре, наклоны по осям X,Y, Z, AUs, эмоции. Каждая строка csv-файла содержит результаты обработки и оригинала кадра, и отражений. Если по какой-то причине кадр не был обработан или не были обнаружены лица, строка, соответствующая этому кадру, просто содержит нулевые значения.

1. Трудности, с которыми пришлось столкнуться
2. Первое затруднение, с которым пришлось столкнуться, заключалась в том, чтобы установить все необходимые пакеты и библиотеки. На [сайте PyFeat](#py_feat_org) есть вся документация по использованию библиотеки, в том числе инструкция по установке, однако она не очень подробная. Проблема заключается в том, что PyFeat поддерживает только три версии языка Python: 3.7, .3.8, 3.9, поэтому сначала пришлось разбираться с тем, чтобы установить нужную версию языка.
3. Было непросто разобраться, как грамотно преобразовать изображения и кадры в PyTorch тензоры, так как надо было учитывать размерность изображений, количество и порядок каналов, разные типы данных. Для корректного анализа было важно, чтобы и кадр, и его тензор выглядели одинаково. Часть вопросов мне удалось решить с помощью [сервиса StackOverflow](#stackoverflow).
4. Потребовалось время, чтобы определить методы вычисления угла наклона головы и вычисления средней линии лица, которые бы давал наиболее точные результаты.

**Заключение**

В ходе практики была успешно разработана программа для хирального анализа лицевой асимметрии на основе видеоданных, а так же изучены все необходимые библиотеки и методы в процессе разработки. Главным итогом работы стало создание функционального прототипа программы, реализующего весь цикл обработки: от загрузки видео до сохранения результатов анализа в CSV файл. Программа выполняет обнаружение лиц, выравнивание изображения по углу наклона головы, построение зеркальных химер и анализ лицевых признаков с использованием библиотеки PyFeat. Результаты анализа, сохраненные в CSV формате, позволяют проводить дальнейший статистический анализ и исследования асимметрии лица. Таким образом, цели практики, связанные с разработкой алгоритма и программы для хирального анализа, были достигнуты.

В процессе работы над проектом я освоила новы инструменты и библиотеки, такие как PyFeat и PyTorch и углубила свои знания в области программирования на Python. Сформированы навыки работы с видеоданными, предобработки изображений и извлечения лицевых признаков. Наибольшие трудности возникли на этапе выравнивания изображения по углу наклона головы и зеркального отображения.

**Список литературы**

1. <https://github.com/cosanlab/py-feat>

1. <https://pytorch.org/tutorials/beginner/basics/intro.html>

1. <https://docs.opencv.org/3.4/d9/df8/tutorial_root.html>
2. https://habr.com

1. <https://py-feat.org>
2. https://stackoverflow.com

**Приложения**

1. Изображение выглядит как текст, снимок экрана, Шрифт

   Автоматически созданное описание
2. Изображение выглядит как зарисовка, Лоб, бровь, Человеческое лицо

   Автоматически созданное описание
3. Список AUs

|  |  |  |
| --- | --- | --- |
| AU1 | Inner brow raise | Подниматель внутренней части брови |
| AU2 | Outer brow raise | Подниматель внешней части брови |
| AU4 | Brow lowerer | Опускатель брови |
| AU5 | Upper lid raise | Подниматель верхнего века |
| AU6 | Cheek raise | Подниматель щеки |
| AU7 | Lids tight | Натягиватель век |
| AU9 | Nose wrinkle | Сморщиватель носа |
| AU10 | Upper lip raise | Подниматель верхней губы |
| AU11 | Nasolabial furrow deepener | Углубитель носогубной складки |
| AU12 | Lip corner puller | Подниматель угла губы |
| AU14 | Dimpler | Втянутый угол губы |
| AU15 | Lip corner depressor | Опускатель угла губы |
| AU17 | Chin raiser | Подниматель подбородка |
| AU20 | Lip stretcher | Растягиватель губ |
| AU23 | Lip tightener | Натягиватель губ |
| AU24 | Lip presser | Сжиматель губ |
| AU25 | Lips part | Губы разделены |
| AU26 | Jaw drop | Челюсть опущена |
| AU28 | Lip suck | Втягивание губ |
| AU43 | Eyes closure | Глаза закрыты |

1. Action units (здесь и далее сокращенно AUs) - это единицы измерения мимических движений, используемые в анализе лицевых действий и выражений эмоций. [↑](#footnote-ref-1)